REMARKS

This Amendment is in response to the Office Action mailed on July 8, 2002.

Claims 1-24 and 35-59 are in this application.

Claims 1-24 and 35-46 are rejected under 35 USC 103(a) as being unpatentable over "Routing on Longest-Matching Prefixes" by Willibald Doeringer et al. (pp. 86-97, 1996 IEEE).

In response, applicants respectfully disagree with the Examiner and argue that for reasons set forth below the claims and newly added claims 47-59 are not obvious in view of the article.

A. INCONSISTENCIES IN THE EXAMINER'S ARGUMENT

To support the rejection the Examiner compares the claims with the article. Pertaining to claim 1, the Examiner seems to argue every element of claim 1 is disclosed in the article. The Examiner then immediately after the comparison states: "Doeringer does not clearly disclose searching string and a table representing a plurality of root nodes of search trees" (page 3 of Office Action).

On one hand the Examiner states that the reference teaches claim 1; on the other hand the Examiner admits a certain element of claim 1 is not clearly disclosed in the reference. It is applicants' contention the reference does not suggest or teach claim 1 and the inconsistency leads the Examiner to the wrong conclusion that applicants' claims are obvious in view of the reference when in fact the claims are not.

B. NOVEL STRUCTURE AND BENEFITS

The Examiner then goes on to summarize what the reference teaches. Applicants assert the Examiner's admission identifying what Doeringer does not teach and his summary of what Doeringer does teach clearly distinguishes the teaching in the article from applicants' claim.

Among the structural difference between applicants' claim and the article is that which the Examiner has admitted on page 3 of the Office Action, i.e. "Searching string and a table representing a plurality of root nodes of search trees". In addition, Doeringer does not teach using a portion of the search string as an address to access the table. By using the table the time required to search the database for a match between a searching string and the database is significantly shortened. For example, the answer could be present in the table at the address whereat the table is accessed. As a consequence only one address would be read in order to find an answer. This is not possible in the Doeringer article since there is no table and the searching is done in a tree structure which is basically a Patricia Tree structure. In contrast, applicants' search structure includes both a table and Patricia Tree coupled to selected entries of the table. It is applicants' contention that the different structure coupled with benefits (i.e. shorter access time to obtain in formation) are clear evidence of unobviousness. Therefore, the claims including the newly added claims are not obvious in view of the teachings of the reference.

In addition, applicants argue that the structure of nodes and method used in walking a tree from the root node to the leaves in applicants' invention is different from that of the article. In applicants' structure a "bird" which is an intermediate leaf is nested in some branches prior to reaching the end leaf. In applicants' invention as expressed in claims 6,

7, 8, 9, 10, 11 and 59 the information which is read from the intermediate nodes are placed on a stack. At the end (claim 59) the pattern stored in the leaf is compared with the search string and the distinguishing position whereat the bit in the search string differ from the stored pattern is used to access the stack to determine the longest prefix match. There is no teaching in the reference equivalent to this feature. In fact, in the article if the search path has to be retracked or retraced then every node along the path is accessed. This requires much longer time to obtain an answer as would be in applicants' claim. In addition, the reference stores both forward and backward pointers. In contrast, applicants store only forward pointers. Storing of forward and backward pointers uses significantly more storage than is required in applicants' invention. The disadvantage of more storage and longer access time to obtain an answer for a particular search string are evidence that applicants' invention is unobvious in light of the teachings of the reference.

The Examiner has compared applicants' claims 2, 3, 4-13, 14 and 15-24 with the reference, but the teaching in applicants' claims and the teachings in the references are so different and distinct that further analysis of the rejection is not warranted. Stated another way the invention claimed by applicants is different from the teachings of the reference.

Notwithstanding, applicants would like to address the Examiner's position relative to claim 3. As to claim 3 the Examiner states "with respect to claim 3, Doeringer discloses where in computer processing device is a network processors (p. 94 under conclusions section-a variety of network functions multiprotocol environment)". Applicants respectfully disagree with the Examiner's position and point out that in the reference there is no mention of the network processor as a device. A variety of network functions and multiple protocol environment are certainly not devices and a teaching of the device being a network processor is unique to applicants' claim and not recited in the reference.

The newly added claims include claims covering the data structure which is in applicants' database. This data structure is the same data structure that is recited in the method claims. This being the case it is the applicants' position that the Examiner's search should already include a data structure. Therefore, adding these claims would not require a new search that would render the need to group these newly added claims as distinct as the Examiner has done with claims 25-34 of the present invention.

It is believed that the present amendment answers all the issues raised by the Examiner. Re-examination is hereby requested and an early allowance of all the claims is solicited.

Attached hereto is a marked-up version of the changes made to the specification and claims by the current amendment. The attached pages are captioned "VERSION WITH MARKINGS TO SHOW CHANGES MADE".

Respectfully submitted,

Joscelyn G. Cockburn
Reg. No. 27,069
Attorney of Record

JGC:ko
Phone: 919-543-9036
FAX: 919-254-2649

VERSION WITH MARKINGS TO SHOW CHANGES MADE
Deletions are shown with brackets [ ] Additions are underscored.

In the Specification:

Page 1, paragraph beginning at line 7 has been amended as follows:

This application is related to, and contains common disclosure with, co-pending and commonly assigned patent applications "Network Processor Processing Complex and Methods", serial number 09/384,691, filed August 27, 1999; "Full Match (FM) Search Algorithm Implementation for a Network Processor", serial number 09/543,531, filed April 6, 2000 [(attorney docket RAL-1999-0139)]; and "Software Management Tree Implementation for a Network Processor", serial number 09/545,1000, filed April 6, 2000 [(attorney docket RAL-1999-0141)]. Each co-pending patent application is hereby incorporated by reference into this description as fully as if here represented in full.

Page 2, paragraph beginning at line 1 has been amended as follows:

The demand for hardware-integrated processing to support more and more complex tasks at media speed has led to the creation of network processors. Network processors provide wirespeed frame processing and forwarding capability with function flexibility through a set of embedded, programmable protocol processors and complementary system coprocessors. Network processors are expected to become the fundamental network building block for [networks] network devices in the manner that microprocessors are for today's personal computers. Network processors offer real-time processing of multiple data streams, providing enhanced security and IP packet handling and forwarding capabilities. In addition, they provide speed improvements through advanced architectures, such as parallel distributed processing and pipeline processing

designs. These capabilities can enable efficient search engines, increased data handling throughput, and provide rapid execution of complex tasks. The programmable features of network processors provide network product developers an easier migration path to implement new protocols and technologies without requiring new custom Application Specific Integrated Circuit (ASIC) designs.

Page 4, paragraph beginning at line 4 has been amended as follows:

A typical system developed with a network processor uses a distributed software model, with each programmable network processor executing tasks concurrently. Some functions are performed in the control point (CP) processor, which can be internal or external to the network processor. The CP processor provides support for layer 2 and layer 3 routing protocols, and layer 4 and layer 5 network applications and systems management. Wirespeed forwarding and filtering functions are performed by a combination of the network processor hardware and resident picocode.

Page 21, paragraph beginning at line 20 has been amended as follows:

The network processor 10 usually resides on a subsystem board and provides the protocol layer(i.e., layer 2, layer 3, layer 4 and higher) frame processing. Software running on a CP processor 34 in the CP subsystem provides the management and route discovery functions. The CP code, picocode running on the protocol processors, and picocode running on the guided frame handler enable initialization of this system, maintenance of the forwarding paths, and management of the system. As a distributed system, the CP processor and each network processor subsystem contain multiple processors which operate in parallel and communicate using guided frames for increased efficiency and performance.

Page 29, paragraph beginning at line 6 has been amended as follows:

For LPM trees, the input key will be hashed into a HashedKey 106, as shown in Fig. [4] 5. In the preferred embodiment, no hash function is performed on the input key for LPM trees, and the hashed output equals the input key. The hash algorithm (including no hash for LPM trees) that will be used is specified in the LUDefTable.

Page 30, paragraph beginning at line 7 has been amended as follows:

If colors are enabled for the tree, which is the case in the example of Fig. [4] 5, the 16-bit color register 124 is inserted in the 176-bit hash function output and the file result is a 192-bit number, called the HashedKey 106. The insertion occurs directly after the direct table 108. If the direct table 108 contains 2^N entries, then the 16-bit color value is inserted at bit position N, as shown in Fig. [4] 5. The output of the hash function, together with the inserted color value, is stored in the HashedKey register 106. If colors are disabled for a tree, the 176-bit hash function is taken unmodified, and 16 zeros are appended to the hash output to produce the 192-bit final HashedKey.

Page 40, paragraph beginning at line 8 has been amended as follows:

When a PSCB is encountered during a search in an LPM tree, the tree search engine hardware 70 will continue the tree-walk on the 0-branch or the 1-branch, depending on the value of the bit p of the UnhashedKey [HashedKey].

In the Claims:

Claims 1 and 35 have been amended as follows:

1. (Amended) A method for determining a longest prefix match for a variable length search key by a computer processing device, comprising the acts of:

reading an input key as a search string;

using the N most significant bits of the input key as an address to index into a table representing a plurality of root nodes of search trees wherein each non-empty entry contains a pointer to a next branch in the search tree or a leaf;

determining if the pointer in a non-empty table entry points to a leaf or a next branch of the corresponding search tree;

reading the next branch contents if the pointer does not point to the leaf of the corresponding search tree and comparing the prefix represented by the next branch with the input key to find a distinguishing bit position;

reading a leaf pattern when the leaf of a corresponding search tree is reached and comparing the leaf pattern with the input key to determine if the leaf pattern matches the input key; and

returning the longest prefix match found for the input key to a requesting application.

35. (Amended) A computer readable medium containing a program product for determining a longest prefix match for a variable length search key, comprising:

program instructions that read an input key as a search string;

program instructions that use the N most significant bits of the input key as

an address to index into a table representing a plurality of root nodes of search trees wherein each non-empty entry contains a pointer to a next branch in the search tree or a leaf;

program instructions that determine if the pointer in a non-empty table entry points to a leaf or a next branch of the corresponding search tree;

program instructions that read the next branch contents if the pointer does not point to the leaf of the corresponding search tree and compare the prefix represented by the next branch with the input key to find a distinguishing bit position;

program instructions that read a leaf pattern when the leaf of a corresponding search tree is reached and compare the leaf pattern with the input key to determine if the leaf pattern matches the input key; and

program instructions that return the longest prefix match found for the input key to the requesting application.

New Claims 47-59 have been added.